PUCRS ESCOLA POLITÉCNICA

Algoritmos de Divisão e Conquista: Multiplicação de Inteiros

O PROBLEMA:

- O problema que consideramos é extremamente básico: a multiplicação de dois números inteiros.
- Este problema é tão básico que inicialmente não podemos pensar nele mesmo como uma questão algorítmica.
- Os alunos do ensino fundamental aprendem um algoritmo concreto (e bastante eficiente) para multiplicar dois números de n dígitos u e v.
- Primeiro você calcula um "produto parcial" multiplicando cada dígito de v separadamente por u e depois soma todos os produtos parciais.

$$\begin{array}{r} 12 \\ \times\ 13 \\ \hline 36 \\ 12 \\ \hline 156 \end{array} \qquad \begin{array}{r} 1100 \\ \times\ 1101 \\ \hline 1100 \\ 0000 \\ 1100 \\ 1100 \\ \hline 10011100 \end{array}$$

(a) **(b)**

- No ensino fundamental sempre vemos isso sendo feito na base 10, mas funciona exatamente da mesma maneira também na base 2.)
- Contar uma única operação em um par de bits como uma primitiva etapa deste cálculo, leva tempo $O(n)$ para calcular cada produto parcial.
- E tempo $O(n)$ para combiná-lo com a soma acumulada de todos os produtos parciais até agora.
- Como existem n produtos parciais, este é um tempo total de execução de $O(n^2)$.
- Se você não pensou muito nisso desde o ensino fundamental, há algo inicialmente surpreendente na perspectiva de melhorar esse algoritmo.
- Todos esses produtos parciais não são "necessários" de alguma forma?
- É possível melhorar o tempo $O(n^2)$ usando uma forma diferente e recursiva de realizar a multiplicação?

Algoritmos de Divisão e Conquista: Multiplicação de Inteiros

PUCRS ESCOLA POLITÉCNICA

O ALGORITMO:

- Sejam u e v dois números com no máximo n dígitos cada.
- Suponha, por enquanto, que n é par.
- Seja p o número formado pelos $n/2$ primeiros dígitos (dígitos mais significativos) de u e seja q o número formado pelos $n/2$ últimos dígitos (dígitos menos significativos) de u. Então:

$$u = p \cdot 10^{n/2} + q$$

- Por exemplo, $u = 2021$, podemos representar como:

$$2021 = 20 \cdot 10^{4/2} + 21$$

- Defina r e s analogamente para v, de modo que:

$$v = r \cdot 10^{n/2} + s$$

- Multiplicando as novas representações de u e v teremos:

$$u \cdot v = \left(p \cdot 10^{n/2} + q\right) \cdot \left(r \cdot 10^{n/2} + s\right)$$

$$= p \cdot r \cdot 10^{n} + p \cdot s \cdot 10^{n/2} + q \cdot r \cdot 10^{n/2} + q \cdot s$$

$$= p \cdot r \cdot 10^{n} + (p \cdot s + q \cdot r) \cdot 10^{n/2} + q \cdot s$$

- O lado direito da equação tem apenas 4 multiplicações verdadeiras (a saber, $p \cdot r$, $p \cdot s$, $q \cdot r$ e $q \cdot s$).
- A multiplicação por 10^n não conta como multiplicação porque representa um mero deslocamento (= shift) do vetor todo para a esquerda.
- Esse deslocamento é muito mais barato que uma multiplicação pois consome meras n unidades de tempo.
- Observação análoga vale para a multiplicação por $10^{n/2}$.

Algoritmos de Divisão e Conquista: Multiplicação de Inteiros

O ALGORITMO:

$$u = p \cdot 10^{n/2} + q$$
$$v = r \cdot 10^{n/2} + s$$

$$u \cdot v = p \cdot r \cdot 10^{n} + (p \cdot s + q \cdot r) \cdot 10^{n/2} + q \cdot s$$

- A expressão acima reduz a multiplicação de dois números com no máximo n dígitos cada a quatro multiplicações de números com no máximo $n/2$ dígitos cada.
- Portanto, temos um primeiro candidato para uma solução de dividir e conquistar:
    - calcular recursivamente os resultados para essas quatro instâncias de $n/2$ bits e depois combiná-los usando a Equação acima.
    - A combinação da solução requer um número constante de adições de números de $O(n)$ bits, portanto leva tempo $O(n)$

- Assim, o tempo de execução $T(n)$ é limitado pela recorrência

$$T(n) \leq 4T(n/2) + cn$$

- para uma constante c.
- Isso é bom o suficiente para nos dar um tempo de execução sub-quadrático?
- Pelo Teorema Mestres, a solução para isso é $T(n) \leq \theta(n^{\log_b a}) = \theta(n^{\log_2 4}) = \theta(n^2)$.

EXEMPLO B: Quero multiplicar 6514202 por 9898989. O resultado das contas pode ser resumido assim:

6514202	$u = p \times 10^4 + q$
9898989	$v = r \times 10^4 + s$
643839	$p \times r$
5851839	$p \times s$
4155778	$q \times r$
37771778	$q \times s$
64484013941778	$p \times r \times 10^8 + (p \times s + q \times r) \times 10^4 + q \times s$

- Ou seja, o algoritmo é mais sofisticado, mas tão lento quando o que aprendemos no colégio...
- Podemos fazer melhor?

Algoritmos de Divisão e Conquista: Multiplicação de Inteiros

<u>O ALGORITMO:</u>

$$u = p \cdot 10^{n/2} + q$$
$$v = r \cdot 10^{n/2} + s$$

$$u \cdot v = p \cdot r \cdot 10^{n} + (p \cdot s + q \cdot r) \cdot 10^{n/2} + q \cdot s$$

- Felizmente, os três números de que precisamos do lado direito da equação — a saber $p \cdot r$, $(p \cdot s + q \cdot r)$ e $q \cdot s$ — podem ser obtidos com apenas três multiplicações.
- Assuma uma nova variável y definida por:

$$y = (p + q) \cdot (r + s)$$

$$= p \cdot r + p \cdot s + q \cdot r + q \cdot s$$

$$y = p \cdot r + p \cdot s + q \cdot r + q \cdot s$$

- Manipulando a equação acima, temos:

$$p \cdot s + q \cdot r = y - p \cdot r - q \cdot s$$

- Portanto a equação $u \cdot v$ pode ser reescrita assim:

$$u \cdot v = p \cdot r \cdot 10^{n} + (y - p \cdot r - q \cdot s) \cdot 10^{n/2} + q \cdot s$$

- É bem verdade que esta equação envolve duas adições e duas subtrações adicionais, mas essas operações consomem muito menos tempo que as multiplicações.
- Se n não for par, basta trocar $n/2$ por $\lceil n/2 \rceil$, o que equivale a acrescentar um 0 à esquerda de u e de v.
- Se denotarmos $\lceil n/2 \rceil$ por m, teremos $u = p \cdot 10^{m} + q$ e $v = r \cdot 10^{m} + s$ e, portanto:

$$u \cdot v = p \cdot r \cdot 10^{2m} + (y - p \cdot r - q \cdot s) \cdot 10^{m} + q \cdot s$$

EXEMPLO C: Quero multiplicar 6514202 por 9898989. O resultado das contas pode ser resumido assim:

06514202	$u = p \times 10^4 + q$
09898989	$v = r \times 10^4 + s$
643839	$p \times r$
4853	$p + q$
9978	$r + s$
48423234	$y = (p + q) \times (r + s)$
10007617	$z = y - p \times r - q \times s$
37771778	$q \times s$
64484013941778	$p \times r \times 10^8 + z \times 10^4 + q \times s$

Algoritmos de Divisão e Conquista: Multiplicação de Inteiros

O ALGORITMO:

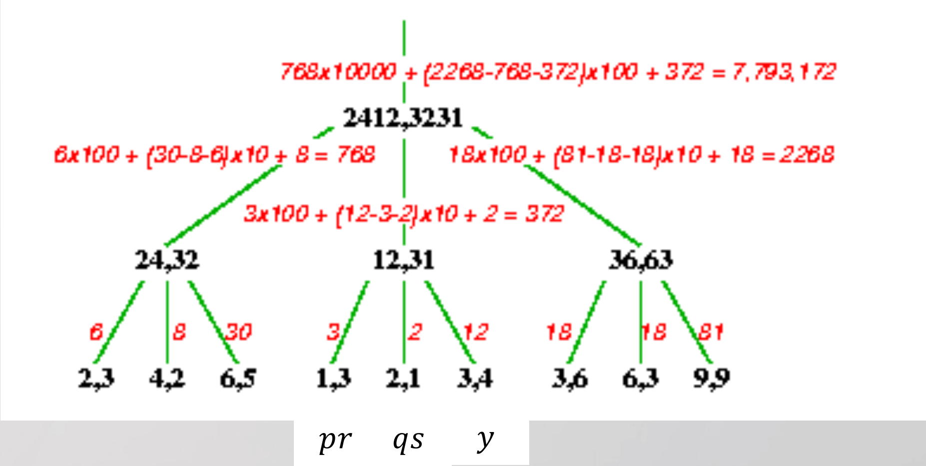
$$u = p \cdot 10^m + q$$
$$v = r \cdot 10^m + s$$

$$u \cdot v = p \cdot r \cdot 10^{2m} + (y - p \cdot r - q \cdot s) \cdot 10^m + q \cdot s$$

- Essas ideias são a base do algoritmo de Karatsuba e Ofman.
- O algoritmo recebe números naturais u e v com no máximo n dígitos cada e devolve o produto $u \times v$:

```
KARATSUBA (u, v, n)
1   se n ≤ 3
2       devolva u×v e pare
3   m := ⌈n/2⌉
4   p := ⌊u/10^m⌋
5   q := u mod 10^m
6   r := ⌊v/10^m⌋
7   s := v mod 10^m
8   pr := KARATSUBA (p, r, m)
9   qs := KARATSUBA (q, s, m)
10  y := KARATSUBA (p + q, r + s, m+1)
11  uv := pr×10^2m + (y − pr − qs)×10^m + qs
12  devolva uv
```

- Na linha 7, $v \bmod 10^m$ é o resto da divisão de v por 10^m, ou seja, o número representado pelos últimos m dígitos da expansão decimal de v.
- As instâncias em que n vale 1, 2 ou 3 devem ser tratadas na base da recursão.
- A linha 11 constitui a fase de combinação do algoritmo.
- A figura abaixo mostra árvore de recursão para a multiplicação de 2412 x 3231.



Algoritmos de Divisão e Conquista: Multiplicação de Inteiros

ANALISANDO O ALGORITMO:

$$u = p \cdot 10^m + q$$
$$v = r \cdot 10^m + s$$

$$u \cdot v = p \cdot r \cdot 10^{2m} + (y - p \cdot r - q \cdot s) \cdot 10^m + q \cdot s$$

- Essas ideias são a base do algoritmo de Karatsuba e Ofman.
- O algoritmo recebe números naturais u e v com no máximo n dígitos cada e devolve o produto $u \times v$:

```
KARATSUBA (u, v, n)
1   se n ≤ 3
2       devolva u×v e pare
3   m := ⌈n/2⌉
4   p := ⌊u/10^m⌋
5   q := u mod 10^m
6   r := ⌊v/10^m⌋
7   s := v mod 10^m
8   pr := KARATSUBA (p, r, m)
9   qs := KARATSUBA (q, s, m)
10  y := KARATSUBA (p + q, r + s, m+1)
11  uv := pr×10^2m + (y − pr − qs)×10^m + qs
12  devolva uv
```

- É claro que o algoritmo produz o resultado correto se $n \leq 3$.
- Suponha agora que $n > 3$. Como $m < n$, lembre-se $m = \lceil n/2 \rceil$, podemos supor, por hipótese de indução, que a linha 8 produz o resultado correto, ou seja, que $pr = p \times r$.
- Analogamente, podemos supor que $qs = q \times s$, $ps = p \times s$ e $qr = q \times r$ no início da linha 11.
- Isso encerra a fase de conquista do algoritmo.
- Assim, o tempo de execução $T(n)$ é limitado pela recorrência

$$T(n) \leq 3T(n/2) + cn$$

- Pelo Teorema Mestre, temos o caso 2:
- $a = 4, b = 2, f(n) = n$
- Então temos que $n^{log_b a} = n^{log_2 3} = n^{1.59} = \Theta(n^{1.59})$
- Como $f(n) = O\left(n^{\log_b a - \epsilon}\right)$, onde $\epsilon = 0.59$, pois $n^{\log_b a - \epsilon} = n^{log_2 3 - 0.59} = n^1 = n$.
- Portanto, $n^{log_b a} > f(n)$, a solução é $T(n) = \Theta(n^{1.59})$, melhor que $\Theta(n^2)$!!!!!